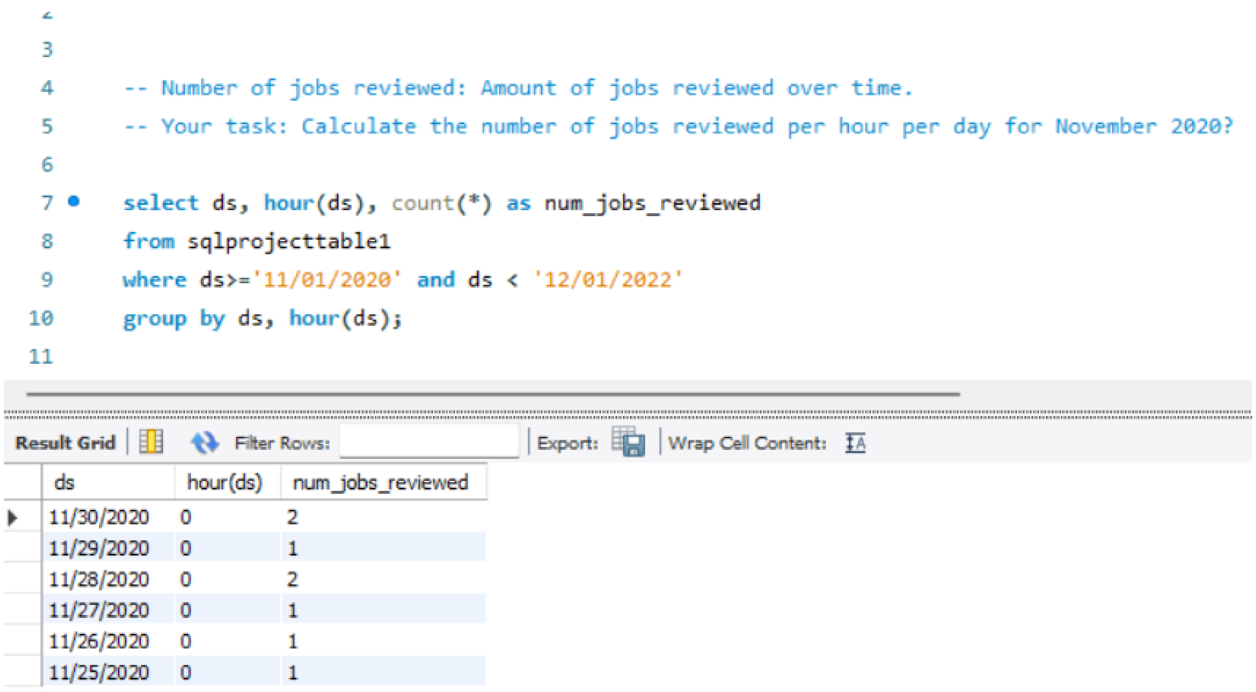
Operation Analytics and Investigating Metric Spike

Case Study 1 (Job Data)

Number of jobs reviewed: Amount of jobs reviewed over time.
Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

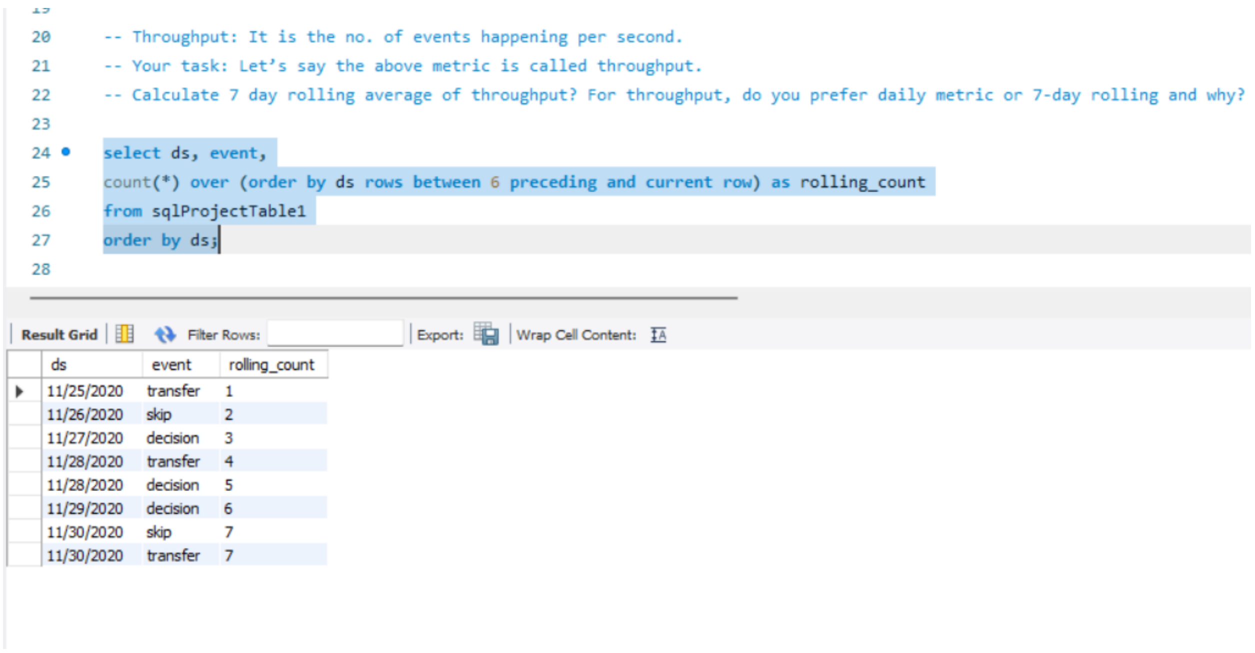


```
-- Number of jobs reviewed: Amount of jobs reviewed over time.
-- Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

select ds, hour(ds), count(*) as num_jobs_reviewed
from sqlprojecttable1
where ds>='11/01/2020' and ds < '12/01/2022'
group by ds, hour(ds);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ds	hour(ds)	num_jobs_reviewed
11/30/2020	0	2
11/29/2020	0	1
11/28/2020	0	2
11/27/2020	0	1
11/26/2020	0	1
11/25/2020	0	1

Throughput: It is the no. of events happening per second.
Your task: Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?



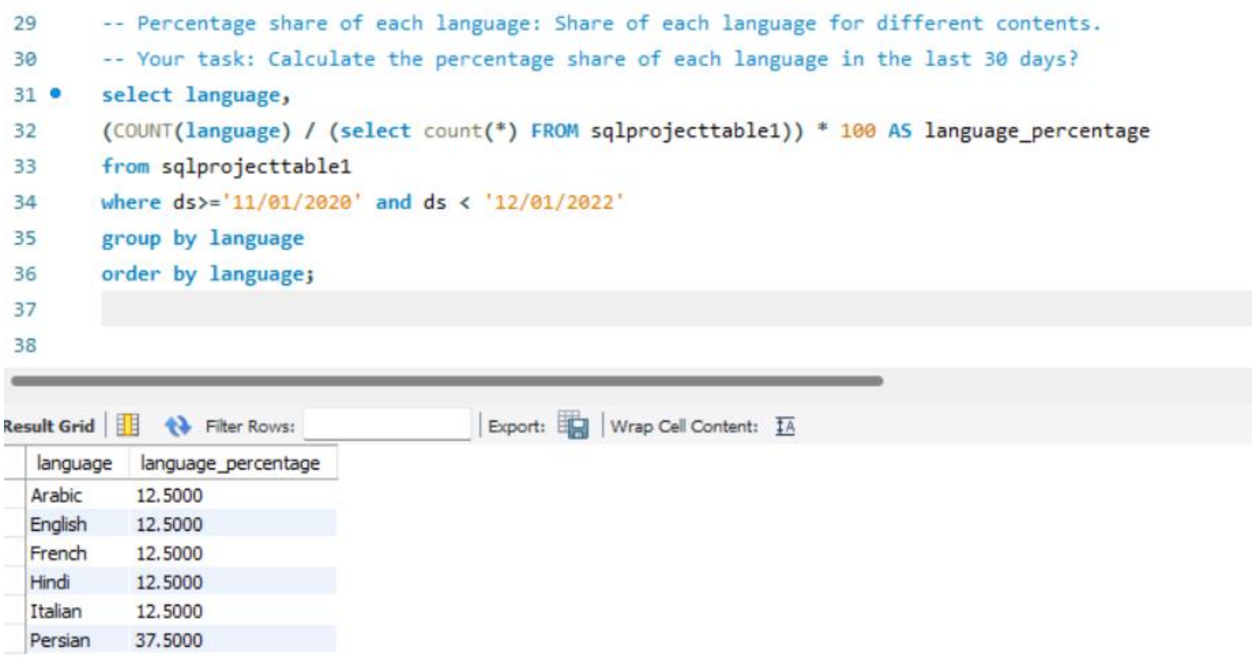
```
-- Throughput: It is the no. of events happening per second.
-- Your task: Let's say the above metric is called throughput.
-- Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?

select ds, event,
count(*) over (order by ds rows between 6 preceding and current row) as rolling_count
from sqlProjectTable1
order by ds;
```

ds	event	rolling_count
11/25/2020	transfer	1
11/26/2020	skip	2
11/27/2020	decision	3
11/28/2020	transfer	4
11/28/2020	decision	5
11/29/2020	decision	6
11/30/2020	skip	7
11/30/2020	transfer	7

For throughput, do you prefer daily metric or 7-day rolling and why?

I would prefer a 7-day rolling metric as we could get more details from the data generated over a week rather than daily generated data, as it might not give generate a large amount of data to analyze.
Organizations which generate tremendous data on daily basis should go for daily throughput metric.

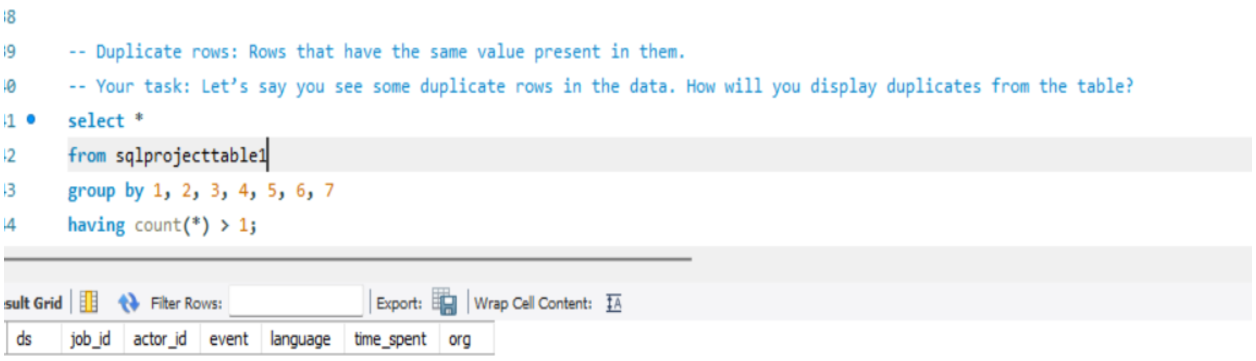
Percentage share of each language: Share of each language for different contents.
Your task: Calculate the percentage share of each language in the last 30 days?



```
-- Percentage share of each language: Share of each language for different contents.
-- Your task: Calculate the percentage share of each language in the last 30 days?
select language,
(COUNT(language) / (select count(*) FROM sqlprojecttable1)) * 100 AS language_percentage
from sqlprojecttable1
where ds>='11/01/2020' and ds < '12/01/2022'
group by language
order by language;
```

language	language_percentage
Arabic	12.5000
English	12.5000
French	12.5000
Hindi	12.5000
Italian	12.5000
Persian	37.5000

Duplicate rows: Rows that have the same value present in them.
Your task: Let’s say you see some duplicate rows in the data. How will you display duplicates from the table?

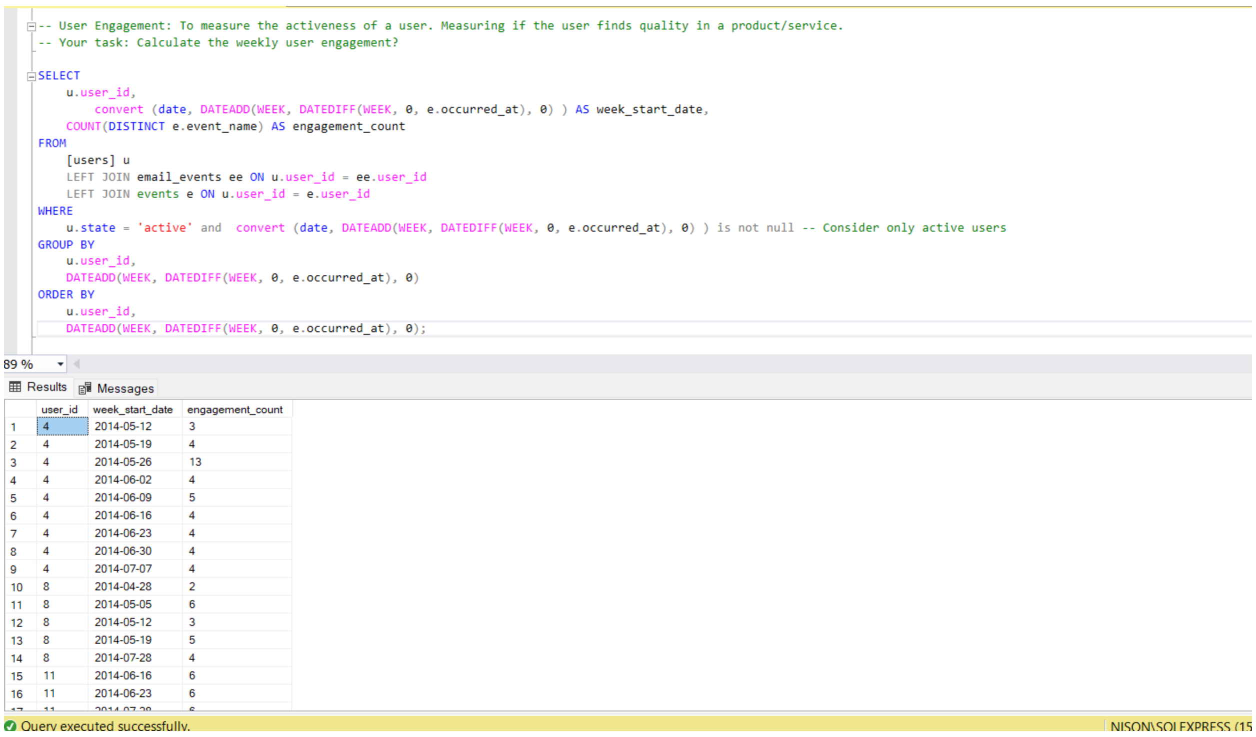


```
-- Duplicate rows: Rows that have the same value present in them.
-- Your task: Let’s say you see some duplicate rows in the data. How will you display duplicates from the table?
select *
from sqlprojecttable1
group by 1, 2, 3, 4, 5, 6, 7
having count(*) > 1;
```

ds	job_id	actor_id	event	language	time_spent	org

Case Study 2 (Investigating metric spike)

User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.
Your task: Calculate the weekly user engagement?



```sql
-- User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.
-- Your task: Calculate the weekly user engagement?

SELECT
    u.user_id,
        convert (date, DATEADD(WEEK, DATEDIFF(WEEK, 0, e.occurred_at), 0) ) AS week_start_date,
    COUNT(DISTINCT e.event_name) AS engagement_count
FROM
    [users] u
    LEFT JOIN email_events ee ON u.user_id = ee.user_id
    LEFT JOIN events e ON u.user_id = e.user_id
WHERE
    u.state = 'active' and  convert (date, DATEADD(WEEK, DATEDIFF(WEEK, 0, e.occurred_at), 0) ) is not null -- Consider only active users
GROUP BY
    u.user_id,
    DATEADD(WEEK, DATEDIFF(WEEK, 0, e.occurred_at), 0)
ORDER BY
    u.user_id,
    DATEADD(WEEK, DATEDIFF(WEEK, 0, e.occurred_at), 0);
```

89 %

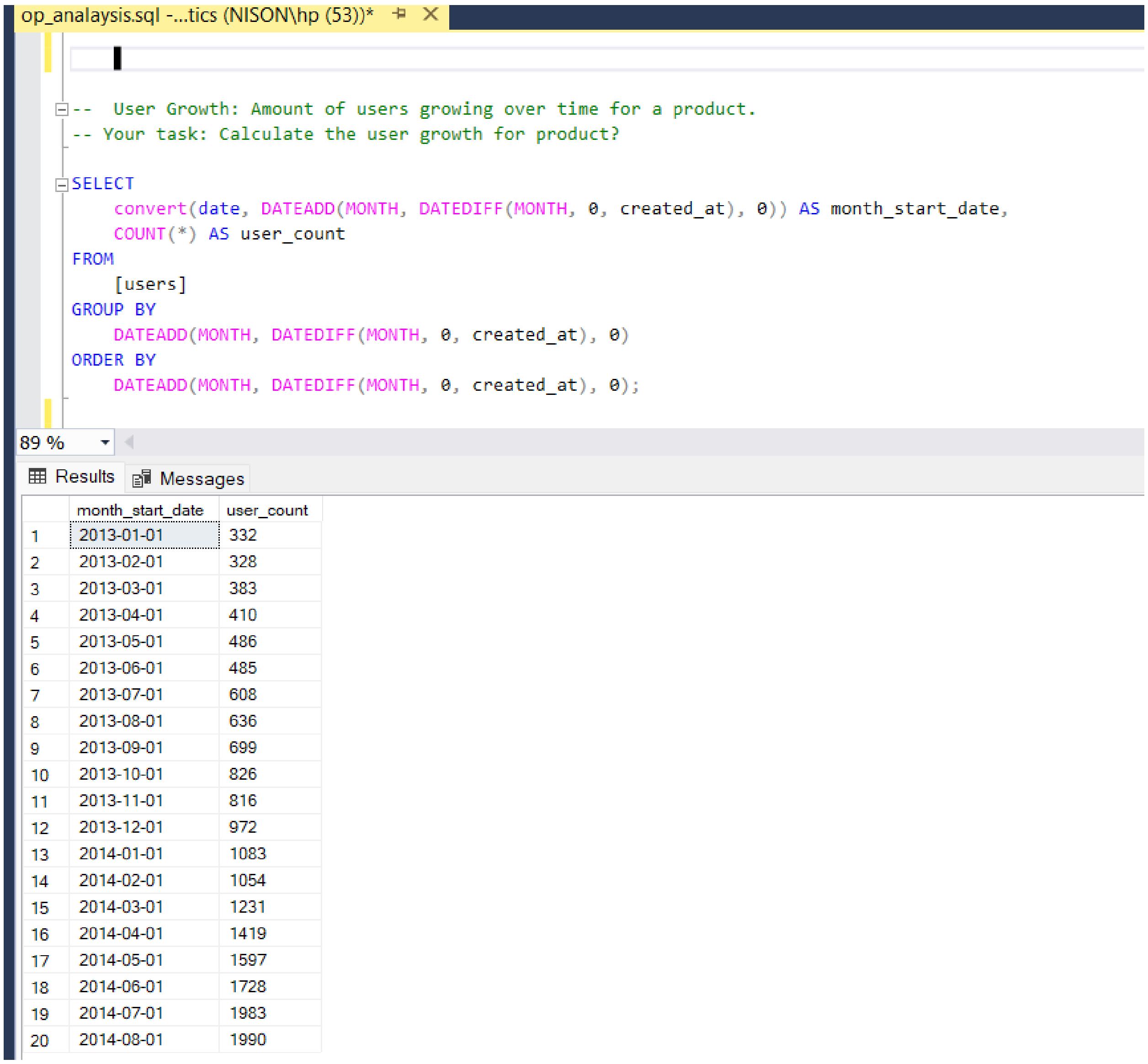
Results | Messages

	user_id	week_start_date	engagement_count
1	4	2014-05-12	3
2	4	2014-05-19	4
3	4	2014-05-26	13
4	4	2014-06-02	4
5	4	2014-06-09	5
6	4	2014-06-16	4
7	4	2014-06-23	4
8	4	2014-06-30	4
9	4	2014-07-07	4
10	8	2014-04-28	2
11	8	2014-05-05	6
12	8	2014-05-12	3
13	8	2014-05-19	5
14	8	2014-07-28	4
15	11	2014-06-16	6
16	11	2014-06-23	6

Query executed successfully.

NISON\SQLEXPRESS (15

User Growth: Amount of users growing over time for a product.
Your task: Calculate the user growth for product?



op_analaysis.sql -...tics (NISON\hp (53))*

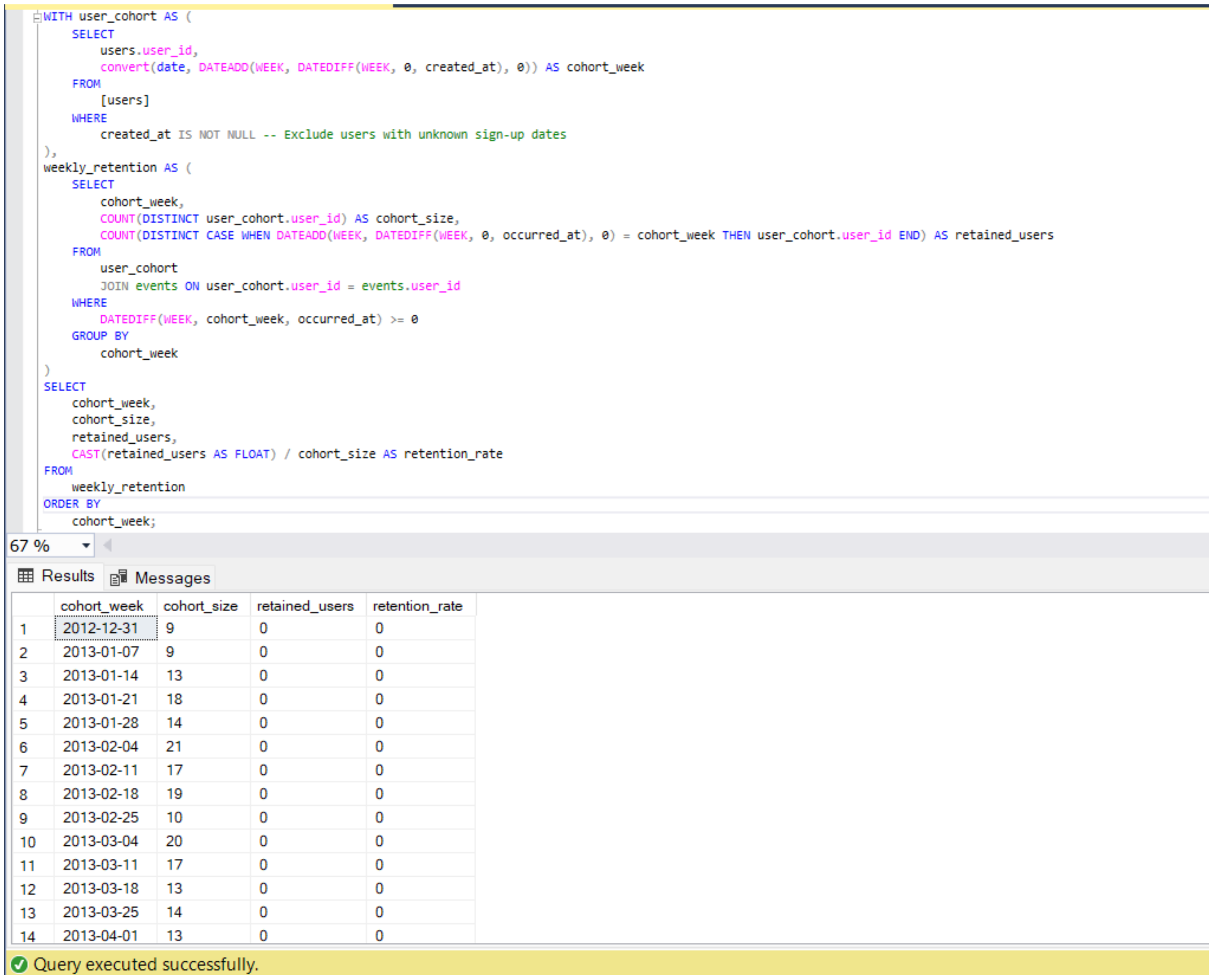
```sql
-- User Growth: Amount of users growing over time for a product.
-- Your task: Calculate the user growth for product?

SELECT
    convert(date, DATEADD(MONTH, DATEDIFF(MONTH, 0, created_at), 0)) AS month_start_date,
    COUNT(*) AS user_count
FROM
    [users]
GROUP BY
    DATEADD(MONTH, DATEDIFF(MONTH, 0, created_at), 0)
ORDER BY
    DATEADD(MONTH, DATEDIFF(MONTH, 0, created_at), 0);
```

89 %

Results | Messages

	month_start_date	user_count
1	2013-01-01	332
2	2013-02-01	328
3	2013-03-01	383
4	2013-04-01	410
5	2013-05-01	486
6	2013-06-01	485
7	2013-07-01	608
8	2013-08-01	636
9	2013-09-01	699
10	2013-10-01	826
11	2013-11-01	816
12	2013-12-01	972
13	2014-01-01	1083
14	2014-02-01	1054
15	2014-03-01	1231
16	2014-04-01	1419
17	2014-05-01	1597
18	2014-06-01	1728
19	2014-07-01	1983
20	2014-08-01	1990

Weekly Retention: Users getting retained weekly after signing-up for a product.
Your task: Calculate the weekly retention of users-sign up cohort?



```sql
WITH user_cohort AS (
    SELECT
        users.user_id,
        convert(date, DATEADD(WEEK, DATEDIFF(WEEK, 0, created_at), 0)) AS cohort_week
    FROM
        [users]
    WHERE
        created_at IS NOT NULL -- Exclude users with unknown sign-up dates
),
weekly_retention AS (
    SELECT
        cohort_week,
        COUNT(DISTINCT user_cohort.user_id) AS cohort_size,
        COUNT(DISTINCT CASE WHEN DATEADD(WEEK, DATEDIFF(WEEK, 0, occurred_at), 0) = cohort_week THEN user_cohort.user_id END) AS retained_users
    FROM
        user_cohort
        JOIN events ON user_cohort.user_id = events.user_id
    WHERE
        DATEDIFF(WEEK, cohort_week, occurred_at) >= 0
    GROUP BY
        cohort_week
)
SELECT
    cohort_week,
    cohort_size,
    retained_users,
    CAST(retained_users AS FLOAT) / cohort_size AS retention_rate
FROM
    weekly_retention
ORDER BY
    cohort_week;
```

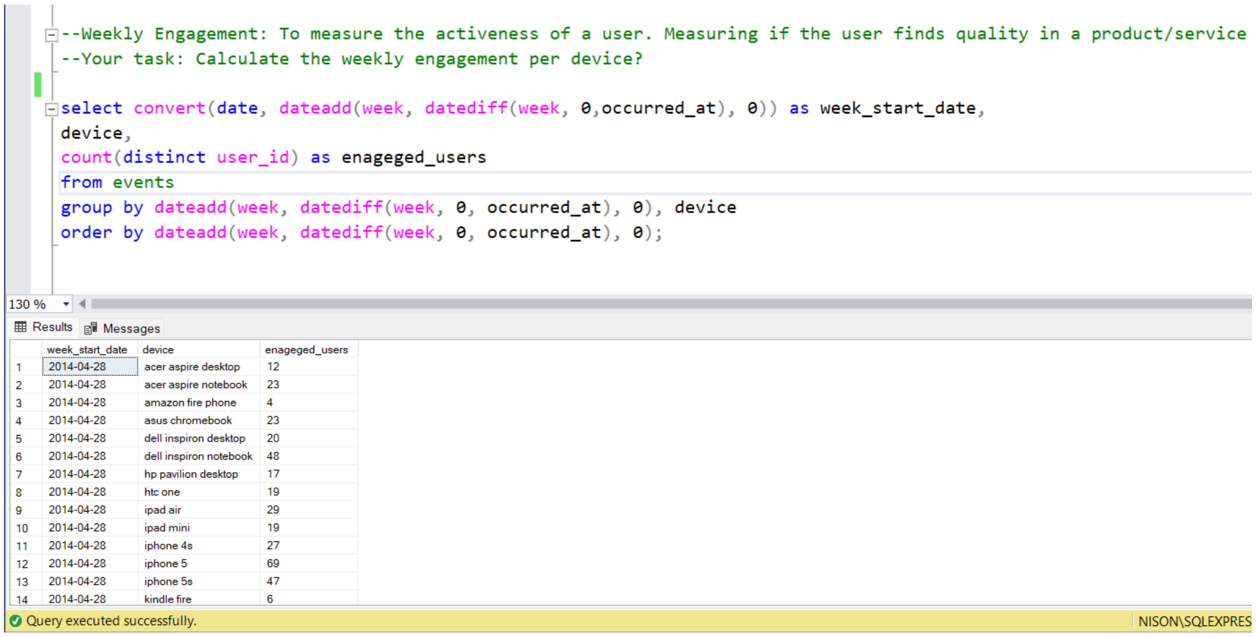
67 %

Results | Messages

	cohort_week	cohort_size	retained_users	retention_rate
1	2012-12-31	9	0	0
2	2013-01-07	9	0	0
3	2013-01-14	13	0	0
4	2013-01-21	18	0	0
5	2013-01-28	14	0	0
6	2013-02-04	21	0	0
7	2013-02-11	17	0	0
8	2013-02-18	19	0	0
9	2013-02-25	10	0	0
10	2013-03-04	20	0	0
11	2013-03-11	17	0	0
12	2013-03-18	13	0	0
13	2013-03-25	14	0	0
14	2013-04-01	13	0	0

Query executed successfully.

Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.
Your task: Calculate the weekly engagement per device?



```
--Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service
--Your task: Calculate the weekly engagement per device?

select convert(date, dateadd(week, datediff(week, 0,occurred_at), 0)) as week_start_date,
device,
count(distinct user_id) as enageged_users
from events
group by dateadd(week, datediff(week, 0, occurred_at), 0), device
order by dateadd(week, datediff(week, 0, occurred_at), 0);
```

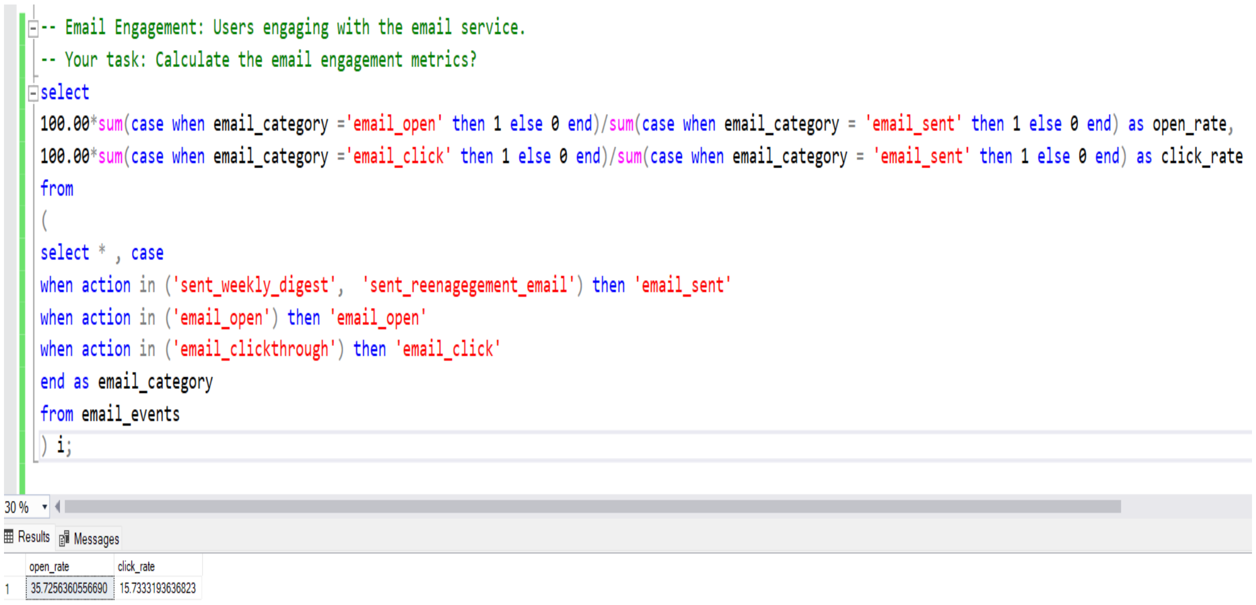
130 %

Results | Messages

	week_start_date	device	enageged_users
1	2014-04-28	acer aspire desktop	12
2	2014-04-28	acer aspire notebook	23
3	2014-04-28	amazon fire phone	4
4	2014-04-28	asus chromebook	23
5	2014-04-28	dell inspiron desktop	20
6	2014-04-28	dell inspiron notebook	48
7	2014-04-28	hp pavilion desktop	17
8	2014-04-28	htc one	19
9	2014-04-28	ipad air	29
10	2014-04-28	ipad mini	19
11	2014-04-28	iphone 4s	27
12	2014-04-28	iphone 5	69
13	2014-04-28	iphone 5s	47
14	2014-04-28	kindle fire	6

Query executed successfully. NISON\SQLEXPRES

Email Engagement: Users engaging with the email service.
Your task: Calculate the email engagement metrics?



```
-- Email Engagement: Users engaging with the email service.
-- Your task: Calculate the email engagement metrics?
select
100.00*sum(case when email_category ='email_open' then 1 else 0 end)/sum(case when email_category = 'email_sent' then 1 else 0 end) as open_rate,
100.00*sum(case when email_category ='email_click' then 1 else 0 end)/sum(case when email_category = 'email_sent' then 1 else 0 end) as click_rate
from
(
select * , case
when action in ('sent_weekly_digest',  'sent_reenagegement_email') then 'email_sent'
when action in ('email_open') then 'email_open'
when action in ('email_clickthrough') then 'email_click'
end as email_category
from email_events
) i;
```

30 %

Results | Messages

	open_rate	click_rate
1	35.7256360556690	15.7333193636823

Project Description:

This project involves analyzing the data from 2 different datasets consisting of 1 tables and 3 tables respectively. It focuses on operational and investigative data analysis approach of data analysis.

Approach:

For this project, I approached the tasks by leveraging SQL queries to retrieve the required data from the provided Instagram database schema. I carefully analyzed the schema to understand the table structures, relationships, and column names. Based on the given tasks, I formulated SQL queries that involved joining tables, aggregating data, and applying filters to extract the necessary information.

Tech-Stack used:

1. SQL: Structured Query Language (SQL) was used to interact with the MySQL database and perform various data manipulation and analysis tasks. SQL queries were used to retrieve specific data, perform aggregations, join tables, and apply filters.

2. MySQL Workbench: MySQL Workbench is a visual tool used for database design, development, and administration. It provides an intuitive interface to connect to the MySQL database, write and execute SQL queries, and visualize the database schema.

3. Microsoft SQL Sever Studio: It is a visual tool used for database design, development, and administration. It provides an intuitive interface to connect to the MySQL database, write and execute SQL queries, and visualize the database schema.

Insights:

Case study 1:

Persian is the language which used by most of the users.

There no duplicate records.

Case Study 2:

Most of the users’ status is ACTIVE.

There is steady growth rate of the product.

Device which is used by most active users is Mac Pro.

Email open rate is more than click rate.

Result:

Overall, this project has enhanced my understanding of data analysis, SQL querying, and deriving meaningful insights from a real-world database. It has strengthened my ability to work with database systems and apply analytical techniques to solve business problems. The project's outcomes have practical implications for marketing campaigns, user engagement, and investor assessments, demonstrating the value of data analysis in decision-making processes.